1. A k-clique in a graph G is a set C of k vertices in G with the property that any two vertices in C are connected by an edge.

Given a graph with n vertices, how many 4 cliques can it have (at most)?

Show that \( L = \{G \mid G \text{ contains a 4-clique}\} \) is in P.

Answer: In a complete graph G with n vertices every set of 4 vertices forms a 4-clique. So the maximum number of 4-cliques for an n vertex graph = (n choose 4) = (n(n-1)(n-2)(n-3))/24 which is \( O(n^4) \).

The polynomial time algorithm for L goes through the \( O(n^4) \) four vertex subsets S of G one at a time, and for each check whether each pair of vertices in S is an edge in G. If yes, then accept. If none of the 4 vertex subsets is a clique then reject.

Each 4 vertex subset of G can be tested as to its being a 4-clique in a constant number of steps, so the whole algorithm can be carried out in time \( O(n^4) \).

2. Now let’s ask the same thing but for n/2 cliques and for n cliques, where n is the number of vertices in the graph G.

i. Specifically, let \( J = \{G \mid G \text{ has n vertices and contains an n/2-clique}\} \) ?

Answer: The same reasoning as in the first question in problem 1 gives the max number of n/2 cliques in G to be \( (n(n-1)(n-2)...(n/2+1))/((n/2)(n/2-1)...(1)) \) which is \( O(2^{n/2}) \).

Can you use the same reasoning as you did in problem 1 to show J is in P?

No. You can use the same algorithm for this as for problem 1 but as there are \( O(2^{n/2}) \) subsets of the vertices of G to check this algorithm is not in polynomial time (relative to n).

ii. Now let \( K = \{G \mid G \text{ has n vertices and contains an n-clique}\} \) ?

Show that K is in P. Answer: To check if G has a clique means to check if all of the n(n-1)/2 pairs of vertices in G are an edge. So just count the edges in G and if all are there then answer yes, else no.
3. Show that P is closed under intersection. That is show that if A and B are sets in P then $A \cap B$ is also in P.

Answer: Our hypothesis is that A is in P, meaning there is an algorithm $A$ which decides A and runs in time $n^c$ for some fixed positive integer c, and that B is in P meaning there is an algorithm $B$ which decides B and runs in time $n^d$ for some fixed positive integer d.

Now the algorithm for $A \cap B$ on input x simply runs the two algorithms $A(x)$ and $B(x)$ accepts x iff both accept. This algorithms runs in polynomial time as it takes at most $|x|^e$ step where $e = \max\{c,d\}$, and so $A \cap B$ is also in P.

4. Prove that TIME $(n^4)$ is closed under difference. That is, if A and B are both in TIME $(n^4)$ then so is A-B.

Our hypothesis is that A is in TIME $(n^4)$, meaning there is an algorithm $A$ which decides A and runs in time $O(n^4)$, and that B is in Time $(n^4)$ meaning there is an algorithm $B$ which decides B and runs in time $O(n^4)$.

Now the algorithm for $A - B$ on input x simply runs the two algorithms on input x and accepts x iff both $A$ and $B$ accept. This algorithms runs in polynomial time as it takes at most $O(|x|^4)$ steps, and so $A - B$ is also in P.

5. Assume that a function f is in polynomial time and can be computed in time $0(n^7)$ and that g is in polynomial time and can be computed in time $0(n^2)$. Prove that f composed with g, that is $f(g(x))$, can be computed in time $0(n^{14})$.

Answer: To compute $f(g(x))$ on an input x of length n you first compute $g(x)$ taking $cn^2$ steps, for some constant c and which outputs a value $g(x)$ whose length is no more than $cn^2$ symbols. Now you compute f on the input $y = g(x)$ to output $f(g(x))$. This takes at most $d|y|^7$ steps.

Noting that $|y| = |g(x)| = cn^2$, computing f(y) takes at most $d|cn^2|^7 = dcn^{14}$ steps. So doing both steps of this computation takes $cn^2 + dcn^{14}$ which is $O(n^{14})$. 